

CALIDAD EN EL DESARROLLO DE SOFTWARE



23 de abril del 2017

Ing. oSCAR JOSUÉ UH PÉREZ, MTGI

5° “B”

yair roberto vega gamboa

Contenido

[Introducción 2](#_Toc480772564)

[Diagrama de Gantt 3](#_Toc480772565)

[Lista de requerimientos 3](#_Toc480772566)

[Funcionales: 3](#_Toc480772567)

[No funcionales: 3](#_Toc480772568)

[Diagrama de Casos de Uso 4](#_Toc480772569)

[Diagrama de Clases 4](#_Toc480772570)

[Cuaderno de registro de Tiempos 5](#_Toc480772571)

[Cuaderno de registro de defectos (logs) 6](#_Toc480772572)

[Estándar Tipos de defectos 6](#_Toc480772573)

[Resumen plan de proyecto 7](#_Toc480772574)

[Código fuente según estándar 8](#_Toc480772575)

[Aplicación ejecutable e instrucciones para ejecutar. 9](#_Toc480772576)

[Conclusión 12](#_Toc480772577)

[Repositorio 12](#_Toc480772578)

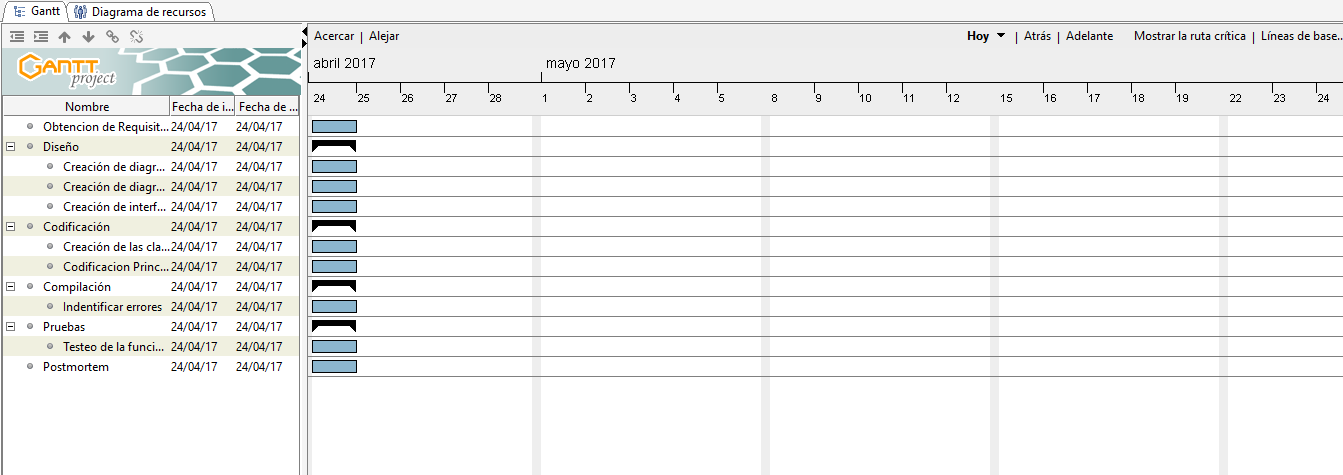
# Introducción

En el presente documento vamos a abordar los principales aspectos del proyecto IMC (Índice de masa corporal), donde se representará las distintas tablas de la estructura PSP0.

En un primer paso se presentarán los puntos acerca el análisis que contiene este proyecto, así como el diagrama de Gantt, sus requerimientos y diagrama de casos de uso.

Después se continúa exponiendo las acciones que cada integrante realizo por medio de las tablas del PSP0, en el que el análisis fue individual y el desarrollo fue por parte de cada uno correspondiendo a las tareas asignadas y en base a los tiempos de control que se manejaron en el controlador de versiones Github.

# Diagrama de Gantt



# Lista de requerimientos

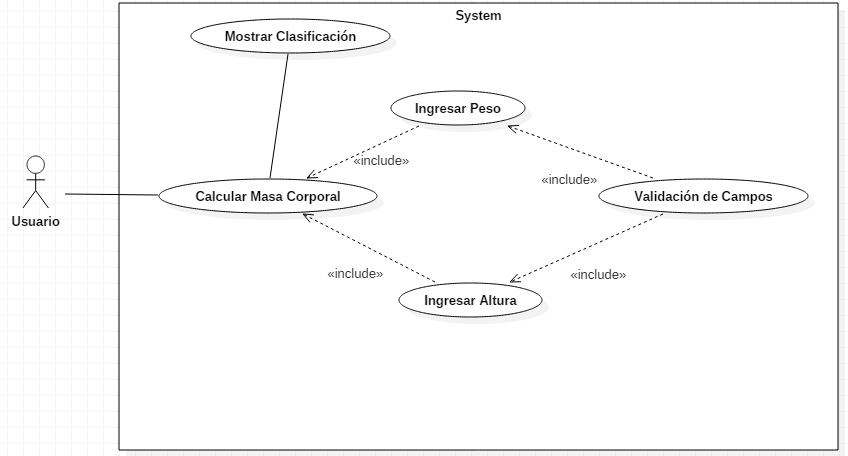
## Funcionales:

* RF01. Obtención del peso y altura por parte del usuario.
* RF02. Validar que los campos de peso y estatura no estén vacíos.
* RF03. Calcular el índice de masa corporal.
* RF04. Clasificación según el índice de masa corporal.

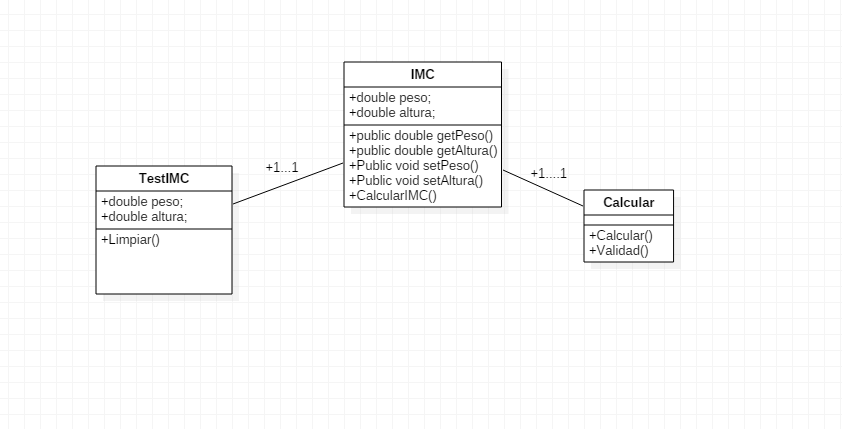
## No funcionales:

* RF02. Validar que se llenen todos los campos

# Diagrama de Casos de Uso



# Diagrama de Clases



# Cuaderno de registro de Tiempos

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Responsable** | **Fecha** | **Inicio** | **Detención** | **Tiempo de Interrupción** | **Tiempo**  **Delta** | **Fase** | **Comentarios** |
| Yair Vega Gamboa | 24/04/2017 | 8:00pm | 8:40pm | 0 minutos | 40 minutos | Análisis | -Creación del diagrama de Gantt  -Creación del diagrama de -caso de uso.  -Creación del diagrama de clases.  -Creación de requerimientos |
| Yair Vega Gamboa | 24/04/2017 | 9:00pm | 9:30pm | 10 minutos | 20  Minutos | Diseño | -Creación de proyecto  -Creación de la interfaz |
| Yair Vega Gamboa | 24/04/2017 | 11:00pm | 11:50pm | 10 minutos | 40  minutos | Desarrollo | -Codificación de la aplicación.  -Codificación de los métodos y clases |
| Yair Vega Gamboa | 24/04/2017 | 11:50pm | 11:59pm | 0 minutos | 9  minutos | Prueba | -Testeo de la aplicación  -Corrección de errores |

# Cuaderno de registro de defectos (logs)

**Yair Roberto Vega Gamboa:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Registro de Defectos | | | | | | |
| Programa: Área de triangulo. | | | | | Fecha:24/04/17 | |
| Fecha | Número | Tipo | Introducido | Eliminado | Tiempo de corrección | Defecto corregido |
| 24/04/17 | 1 | 20 | Sintaxis | Revisión | 5min |  |
| Descripción: | Ortografía Errónea en los comentarios de la clase IMC | | | | | |
| 24/04/17 | 2 | 70 | Datos | Análisis | 10min |  |
| Descripción: | Codificación errónea en la aplicación | | | | | |
| 24/04/17 | 3 | 20 | Sintaxis | Revisión | 5min |  |
| Descripción: | Ortografía Errónea en los comentarios de la aplicación | | | | | |

# Estándar Tipos de defectos

|  |  |  |
| --- | --- | --- |
| **Tipos de defectos** | | |
| **Nº de tipo** | **Nombre del tipo** | **Descripción** |
| 10 | Documentación | Comentarios, mensajes |
| 20 | Sintaxis | Ortografía, puntuación, erratas, formato de las instrucciones |
| 30 | Construir, paquetes | Gestión del cambio, librerías, control de versión |
| 40 | Asignación | Declaración, nombres duplicados, ámbito, límites |
| 50 | Interfaz | Llamadas a procedimientos y referencias, E/S, formatos de usuario |
| 60 | Chequeo | Mensajes de error, chequeos inadecuados |
| 70 | Datos | Estructura, contenido |
| 80 | Función | Lógica, punteros, bucles, recursión, computación, defectos de la función |
| 90 | Sistema | Configuración, temporización, memoria |
| 100 | Entorno | Diseño, compilación, pruebas y otros problemas que soporta el sistema |

# Resumen plan de proyecto

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Programa:** Área de un triángulo | | | | | | **Fecha:** 24/04/2017 | |
| **Descripción:** Software encargado de calcular el índice de masa corporal de un usuario.  Studio. | | | | | | | |
| **Resumen** | **Plan** | | **Real** | | | **Hasta la fecha** | |
| Minutos/LOC | 1 | | 3 | | | 4 | |
| LOC/Hora | 60 | | 160 | | | 200 | |
| Defectos/KLOC | - | | 30 | | | 20 | |
| Rendimiento | - | | - | | |  | |
| Valoración/Fallo | - | | - | | |  | |
| **Tamaño programa (LOC)** | **Plan** | | **Real** | | | **Hasta la fecha** | |
|  | |  | | |  | |
| Total nuevo & cambiado | 58 | | 200 | | | 210 | |
| Tamaño máximo | 100 | |  | | | | |
| Tamaño mínimo | 50 | |
| **Tiempo por Fase (min.)** | **Plan** | **Real** | | **Hasta la fecha** | **% Hasta la fecha** | | |
| Planificación | 30 | 50 | | 42 | 15.8% | | |
| Diseño | 25 | 30 | | 30 | 11.3% | | |
| Codificación | 40 | 55 | | 53 | 20% | | |
| **Revisión del código** |  | | | | | | |
| Compilación | 10 | 25 | | 20 | 7.5% | | |
| Pruebas | 30 | 20 | | 20 | 7.5% | | |
| Postmorten | 40 | 50 | | 90 | 36.7% | | |
| Total | 175 | 230 | | 265 | 98.20% | | |
| Tiempo máximo |  |  | | | | | |
| Tiempo mínimo |  |
| **Defectos Introducidos** | **Plan** | **Actual** | | **Hasta la fecha** | **% Hasta la fecha** | | **Def./Hora** |
| Planificación | - | 50 | | 50 | 10% | | 0.2 |
| Diseño | - | 10 | | 10 | 15% | | 0.1 |
| Codificación | - | 15 | | 15 | 20% | |  |
| Revisión del código | - | 30 | | 30 | 22% | |  |
| Compilación | - | 20 | | 20 | 25% | |  |
| Pruebas | - | 10 | | 10 | 27% | |  |
| Total | - | 135 | | 135 | 119% | |  |
| **Defectos eliminados** | **Plan** | **Actual** | | **Hasta la fecha** | **% Hasta la fecha** | | **Def./Hora** |
| Planificación | - | 45 | | 45 | 10% | |  |
| Diseño | - | 7 | | 7 | 17% | |  |
| Codificación | - | 12 | | 12 | 21% | |  |
| Revisión del código | - | 25 | | 25 | 23% | | 1.5 |
| Compilación | - | 15 | | 15 | 27% | | 1.8 |
| Pruebas | - | 8 | | 8 | 30% | | 2.3 |
| Total | - | 112 | | 112 | 128 | |  |

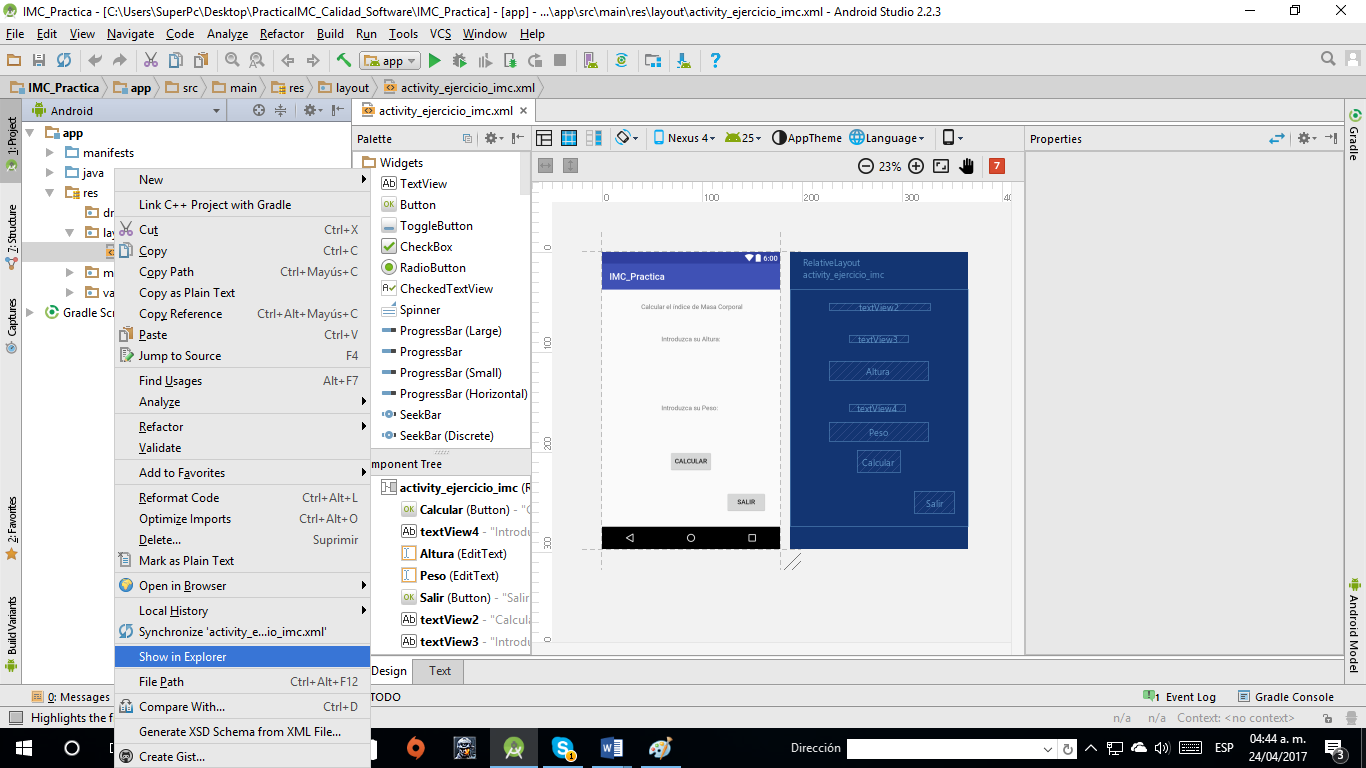
# 

# Código fuente según estándar

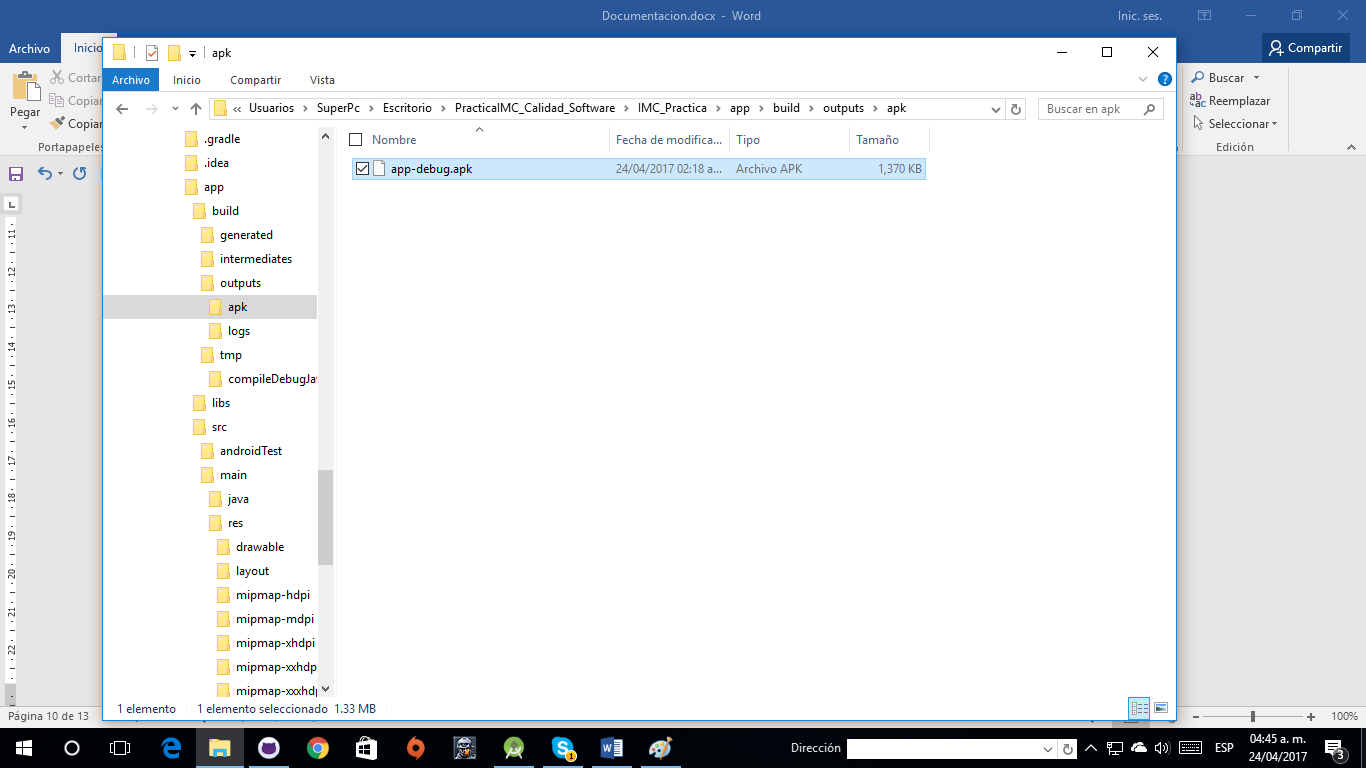
|  |  |  |
| --- | --- | --- |
| Categoría | Descripción | Ejemplo |
| 1 | Métodos | CalcuIMC() |
| 2 | Comentarios | // para comentar |
| 3 | Variables y constantes | peso, altura; |
| 4 | Clases | EjercicioIMC.java |

1. Ejemplo Métodos: Public void CalcuIMC(){}
2. Ejemplo Comentarios: EditText peso; // Variable en la codificación
3. Ejemplo Variables: EditTex peso;
4. Ejemplo Clases: public class EjercicioIMC extends AppCompatActivity {}

# Aplicación ejecutable e instrucciones para ejecutar.

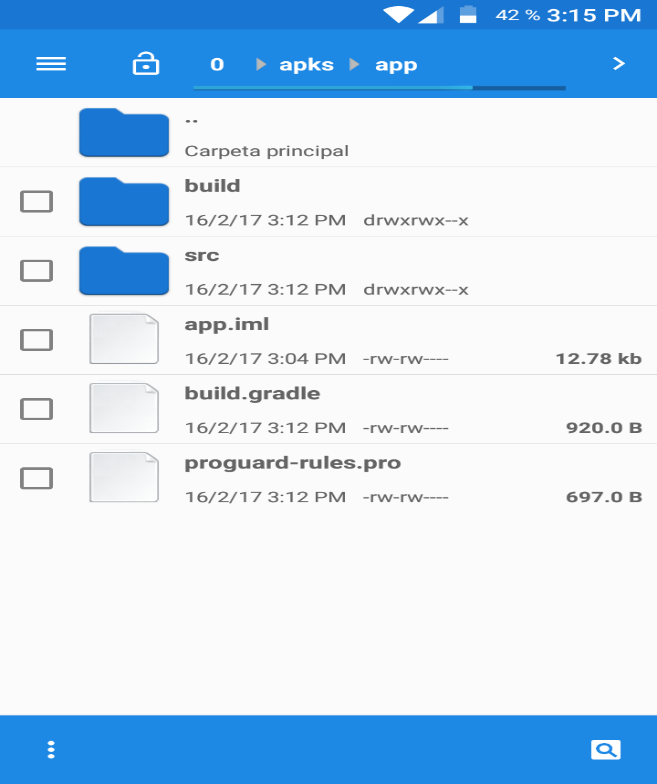


Paso 1: Como primer paso debemos presionar clic derecho a la carpeta app como se muestra en la imagen, seguido de esto aparecerá un menú de opciones donde debemos buscar la sección de “Show in Explorer” y darle clic.



Paso 2: Después del paso 1 se visualizara el explorador de archivos donde se encuentra la carpeta del proyecto, buscamos la carpeta llamada “app” y la copiamos y pegamos en algún lugar que nosotros queremos dentro de la memoria interna del dispositivo Android.

Nota: Debes tener conectado tu teléfono celular a la computadora para realizar el copiado y pegado de la carpeta “app”.

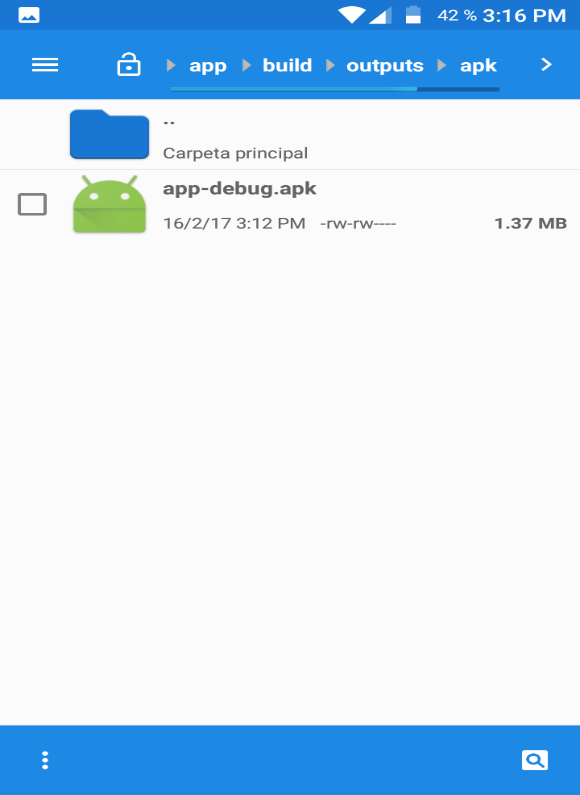
Paso 3: Ya en el dispositivo móvil, debemos encontrar el donde guardamos la carpeta “app”, una vez encontrada debemos entrar a la carpeta “build” seguido de esto aparecerá una seria de carpeta encontrar la carpeta llamada “outputs” e ingresar a ella.

Paso 4: Después de eso aparecerán otras carpetas más nosotros ingresaremos a la carpeta “apk”.

Paso 5: Después de realizar los anteriores pasos de forma adecuada veremos un archivo en dentro de la carpeta “apk” llamado:

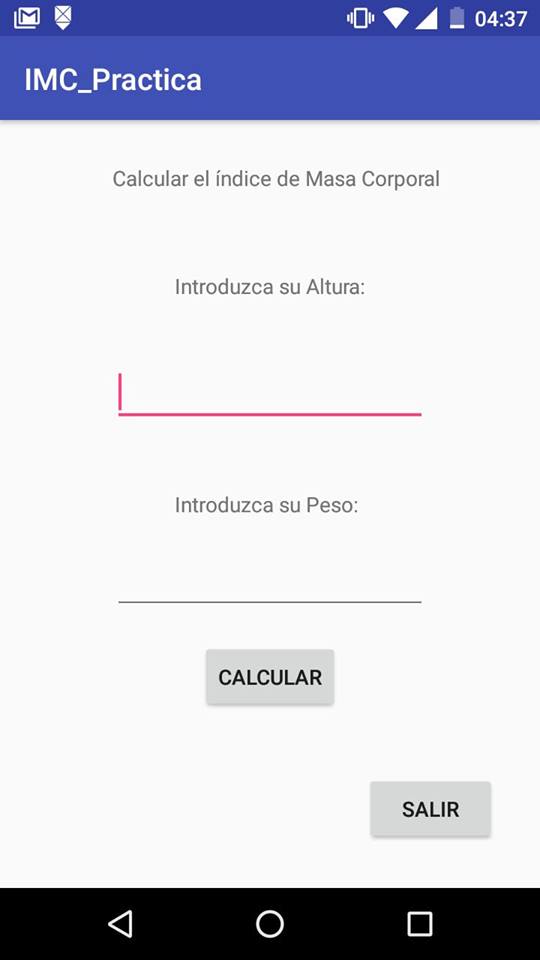
app-debug.apk

Este es el nombre del archivo a instalar en el dispositivo.



Paso 6: Cuando le demos clic al archivo app-debug.apk se visualizará una pantalla de instalar aplicación, el usuario tendrá que instalar dicha aplicación y luego se le abrirá la aplicación del IMC.

Paso 7: Se muestra la aplicación instalada en el dispositivo. Con el diseño que se le ha otorgado por los desarrolladores.



# Conclusión

**Yair Roberto Vega Gamboa:**

El PSP me sirvió para el trabajo individualmente y fue efectiva porque nos muestra una manera más eficiente de cómo controlar los tiempos y la gestión de errores, que tiene cada miembro del equipo y en base a eso nos ayuda a subir la productividad a la hora desarrollo de software. En mi opinión las tablas que maneja el PSP para el control de tiempos nos ayuda mucho a medir que productivos fuimos individualmente en ese tiempo transcurrido y como fuimos desarrollando la aplicación. La tabla de registro de errores fue la que más le busque efectividad porque nos muestra que defectos o errores cometemos individualmente y nos sirve en un futuro para analizar en que estamos “fallando” más y estar más pendiente de esos errores.

# Repositorio

Link : <https://github.com/Vegayair/PracticaIMC_Calidad_Software>